PRODUCT INFORMATION MANAGEMENT

In a dynamic online shop, there must be a way of merchants adding/updating products in the backend. In this assignment we would like to see two separate microservices serving this purpose: one to read products data from a CSV file (Importer) and wrap them into a JSON object for the other service to consume; the second to save and maintain the product data received and provide some statistics about them (Aggregator).

The first service (the Importer) would be a simple stateless service that reads a CSV file containing all the data regarding products and sends them as messages on a message broker of your choice or using restful calls.

The second service (the Aggregator) will handle incoming product messages to create a new product if it doesn't exist in its database, and update it if it was already there.

The Aggregator should be capable of:

- Providing an endpoint to list all available products in its database.
- Providing an endpoint to show daily statistics of how many products were created and how many updated.
- Handling high message load and back pressure.
- Concurrency, which means being aware if the product undergoes simultaneous updates.

A product would have a list of SKUs (stock-keeping unit identifiers) to identify variants of the product (for instance, T-shirts in different sizes or colours), and updates to the product would involve adding or deleting from this list.

Handle any edge case that comes to mind and make sure that this service can run in a containerized environment.

Feel free to use docker compose for running external dependencies (eg. message brokers or databases).

HINTS:
Besides the application being able to run, please keep in mind, that we also would like to see that you have thought about the following:

— How to handle concurrent requests
— Is the backend always in a valid state?
— What about race conditions?
— Is your code tested; have you considered quality measures?
— Have you thought about edge cases?
— Can your project be built without any errors?

HOW TO SUBMIT THE TASK

Please send back the completed task as a git bundle (data format uploaded per email as answer to our email conversation). You can choose whether or not to include your commit history.

https://git-scm.com/book/en/v2/Git-Tools-Bundling

PS: A sample CSV file will be provided with the product data to be consumed.